UNITED STATES PATENT AND TRADEMARK OFFICE

A

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/785,625	02/16/2001	Christine Michelle Barnes	343355600026	4951

7590 08/11/2005

John V. Biernacki
Jones Day, Reavis & Pogue
North Point
901 Lakeside Avenue
Cleveland, OH 44114

EXAMINER	
KANG, INSUN	

ART UNIT	PAPER NUMBER
2193	

DATE MAILED: 08/11/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/785,625	BARNES ET AL.	
	Examiner	Art Unit	
	Insun Kang	2193	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *26 May 2005*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-36* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-36* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is in response to the RCE amendment filed 5/26/2005.

2. As per applicant's request, claims 1, 7, 8, 11, 26, 27, and 32 have been amended. Claims 1-36 are pending in the application.

Claim Rejections - 35 USC § 112

3. The rejection to claims 1-25, 27, and 28 has been withdrawn due to the amendment to the claims.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. Claims 1-36 are rejected under 35 U.S.C. 103(a) as being unpatentable over Johnson (XML JavaBeans, Part 1-3, 2-7/1999, JavaWorld) in view of JSX (JSX history, 11/2000).

Per claim 26:

Johnson discloses generating a node tree whose nodes store the public object state data ("use XML as a serialization format for beans," page 7, Beans as XML documents; "an XML document in a program can be represented as a tree of ...(DOM) nodes...flatten that tree of DOM tree and write it to a text file," pg 5-6 in Part 2; Simpler XML in Part 3; "call the method getAsDOM(), which builds a DOM document tree based

on the properties of the JavaBean," pg 7 par 4-5 in Part 2; see pg 6 par 4-5 in Part1; pg 9 paragraph 4-5 in Part 3).

Johnson does not explicitly teach storing the private object state data wherein the private state data is an object's internal state data. However, JSX teaches serialization of private data was known in the art of software development, at the time applicant's invention was made, to restore the complete object state ("serialization: access to private...fields," page 1 JSX0.5). As the applicant acknowledged in remark filed on 3/7/2005, the private data, which is an object's internal state data, is accessible via the object's public interface, if made accessible as described in Java specification (remark, page 14). Therefore, it would have been obvious for one having ordinary skill in the art of computer software development to modify Johnson's disclosed system to persist the private object state data along with the public data. The modification would be obvious because one having ordinary skill in the art would be motivated to reconstitute the original object state from the serialized form as taught by JSX.

Johnson further discloses:

-the public state data is data that can be retrieved via public method calls or public fields ("JavaBeans," summary)

-wherein an object is queried with respect to its private object state in order to determine the private object state data (see the rationale above)

-processing the nodes of the node tree to generate nodes in an XML tree, wherein the nodes in the XML tree correspond to an XML tag structure ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a

property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6; convert a JavaBean to a tree, and then convert the tree to text, which is then written to a file...The XML correspoinding to the DOM tree," pg 5-6 in Part 2; see Figure 3 in Part 2; pg 8 in Part 1)

-generating XML tags based upon the nodes in the XML tree, wherein the XML tags are structured so as to persist the public and private object state data (see Figures 3 and 4, Dom tree for the JavaBean and XML representing the DOM tree in Part 2, page 6; XML as a persistence format for JavaBeans components," pg 2 in Part 2)

Per claim 27:

The rejection of claim 26 is incorporated, and further, Johnson teaches:

- parsing the XML tags to recover the public and private object state data ("the XML parser the name of the XML file, and it returned the entire data structure that the XML file represented," page 3, Part 2; readXMLBean ...creates an XML parser, reads the input the input XML file," pg 12 par 3 in Part 3)
- instantiating objects based upon the recovered public and private object state data (i.e. "instantiateBean() in page 9; "It uses XML to instantiate a JavaBean and set that bean's properties for a JavaBean class that already exists," page 7, Part 1; "identities the first element of the document read, and calls instantiateBean() to create an instance of the bean and initialize it. This method parses XML into a DOM tree by calling a canned parser," pg 12 par 3-4 in Part 3)

- using the instantiated objects within the object development environment ("It uses XML to instantiate a JavaBean and set that bean's properties for a JavaBean class that already exists," page 7, Part 1; see Listing 4. readXMLBean() parses XML and calls instantiateBean() in Part 3; "controlling XMLBeans properties by integrating XMLBeans with the core java.beans package," pg 2 par 1 in Part 3)
- the public state data is data that can be retrieved via public method calls or public fields; the private state data is an object's internal state data (page 7, Part 1) as claimed.

Per claim 28:

The rejection of claim 27 is incorporated, and further, Johnson teaches:

- -parsing the XML tags to recover the design time object state data ("the XML parser the name of the XML file, and it returned the entire data structure that the XML file represented," page 3, Part 2)
- -instantiating objects based upon the recovered design time object state data ("It uses XML to instantiate a JavaBean and set that bean's properties for a JavaBean class that already exists," page 7, Part 1)
- -using the instantiated objects within the object development environment such that the recovered design time object state data is used only within the object development environment ("It uses XML to instantiate a JavaBean and set that bean's properties for a JavaBean class that already exists," page 7, Part 1) as claimed.

Per claim 29:

The rejection of claim 26 is incorporated, and further, Johnson teaches:

- parsing the XML tags to recover the public and private object state data ("It uses XML to instantiate a JavaBean and set that bean's properties for a JavaBean class that already exists," page 7, Part 1)

-generating source code based upon the recovered public and private object state data (i.e. see Creating JavaBeans from XML, page 9, part 1)

-using the generated source code to perform a computer operation (i.e. see Creating JavaBeans from XML, page 9, part 1) as claimed.

Per claim 30:

The rejection of claim 26 is incorporated, and further, Johnson teaches that the public and private object state data comprise state data from JavaBeans (JavaBean, summary) as claimed.

Per claim 31:

The rejection of claim 26 is incorporated, and further, Johnson teaches:

- parsing the XML tags to recover the public and private object state data ("the XML parser the name of the XML file, and it returned the entire data structure that the XML file represented," page 3, Part 2)

- instantiating objects in an order based upon the stored state restoration order wherein the instantiating of the object recovers the public and private object state data ("It uses

XML to instantiate a JavaBean and set that bean's properties for a JavaBean class that already exists," page 7, Part 1)

-using the instantiated objects within the object development environment ("It uses XML to instantiate a JavaBean and set that bean's properties for a JavaBean class that already exists," page 7, Part 1).

Per claim 1:

Johnson discloses:

- determining the private object state data of objects used within the object development environment ("Identify the JavaBean's property names, types and values... Instrospector and ...PropertyDescriptor," page 3, part 2; "XML file that specifies the values for a JavaBeans' properties," pg 7 paragraphs 2-8; pg 9 last paragraph –first paragraph in pg 10 in Part 3)

-an object is queried with respect to its private object state in order to determine the private object state data wherein the private state data is an object's internal state data, and public state data is data that can be retrieved via public method calls or public fields ("Identify the JavaBean's property names, types and values... Instrospector and ...PropertyDescriptor," page 3, part 2).

Johnson discloses storing public data in a human readable file, XML file. However, he does not explicitly teach storing the determined private object state data in a computer-readable file wherein the computer-readable file is in a human-understandable format. However, JSX teaches serialization of private data was known

in the art of software development, at the time applicant's invention was made, to restore the complete object state ("serialization: access to private...fields," page 1 JSX0.5). As the applicant acknowledged in remark filed on 3/7/2005, the private data, which is an object's internal state data, is accessible via the object's public interface, if made accessible as described in Java specification (remark, page 14). Therefore, it would have been obvious for one having ordinary skill in the art of computer software development to modify Johnson's disclosed system to persist the private object state data along with the public data. The modification would be obvious because one having ordinary skill in the art would be motivated to reconstitute the original object state from the serialized form as taught by JSX.

Johnson further discloses:

Restoring the private object state data by processing the computer-readable file ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6) as claimed.

Per claim 2:

The rejection of claim 1 is incorporated, and further, Johnson teaches that the human-understandable format is a text-based format ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6) as claimed.

Per claim 3:

The rejection of claim 1 is incorporated, and further, Johnson teaches:

- after storing the determined private object state data in the computer-readable file, modifying the private object state data within the computer-readable file without using the object development environment (i.e. Customization, page 2, part 3)
- restoring the stored private object state data by processing the computer-readable file, wherein the restored private object state data contains the modifications to the private object state data (i.e. Customization, page 2, part 3)

Per claim 4:

The rejection of claim 1 is incorporated, and further, Johnson teaches:

-after storing the determined private object state data in the computer-readable file, directly editing the computer-readable file in order to modify the private object state data within the computer-readable file(i.e. Customization, page 2, part 3)

-restoring the stored private object state data by processing the computer-readable file, wherein the restored private object state data contains the modifications to the private object state data (i.e. Customization, page 2, part 3) as claimed.

Per claim 5:

The rejection of claim 1 is incorporated, and further, Johnson teaches that the modifications are to correct errors in object structure without using the object development environment (i.e. Customization, page 2, part 3).

Per claim 6:

The rejection of claim 1 is incorporated, and further, Johnson teaches:

- an object class that specifies structure of the private object state data has been modified (i.e. Customization, page 2, part 3)

-said modification occurring after the private object state data has been stored in the computer-readable file (i.e. Customization, page 2, part 3)

-restoring the private object state data from the computer-readable file even though the object class has been modified (i.e. Customization section, page 2, part 3).

Per claim 7:

The rejection of claim 1 is incorporated, and further, Johnson teaches:

- an object class that is included in the object development environment and that specifies structure of the private object state data has been modified (pg 2 Part 3, Customization section)

-said modification occurring after the private object state data has been stored in the computer-readable file (pg 2 Part 3, Customization section)

-restoring back into the object development environment substantially the private object state data from the computer-readable file despite the structures differing between the modified class and the private object state data (pg 2 Part 3, Customization section)

Per claim 8:

The rejection of claim 1 is incorporated, and further, Johnson teaches:

-determining public and the private object state data of the objects used within the object development environment ("Identify the JavaBean's property names, types and values...Instrospector and ...PropertyDescriptor," page 3, part 2)

- storing the determined public and private object state data in the computer-readable file (see the rationale above)

-restoring the private and public object state data by processing the computer-readable file ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6) as claimed.

Per claim 9:

The rejection of claim 1 is incorporated, and further, Johnson teaches a Java development environment ("JavaBeans," summary).

Per claim 10:

The rejection of claim 1 is incorporated, and further, Johnson teaches a Java development environment for providing graphical user interfaces ("JavaBeans," summary).

Per claim 11:

The rejection of claim 1 is incorporated, and further, Johnson teaches determining that first private object state data is to be restored before second private object state

data...computer-readable file is in a structured format that indicates order in which the private object state data is to be restored ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6) as claimed.

Per claim 12:

The rejection of claim 11 is incorporated, and further, Johnson teaches:

restoring the private object state data in the order specified by the structured format of the computer-readable file ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6) as claimed.

Per claim 13:

The rejection of claim 12 is incorporated, and further, Johnson teaches that the structured format is an XML structured format ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6) as claimed.

Per claim 14:

The rejection of claim 13 is incorporated, and further, Johnson teaches that the XML structured format includes nested XML blocks to indicate the order in which the private

object state data is to be restored ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6) as claimed.

Per claim 15:

The rejection of claim 1 is incorporated, and further, Johnson teaches that the computer-readable file is in a structured format that contains private and public object state data (see the rejection of claim 1; "creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6) as claimed.

Per claim 16:

The rejection of claim 1 is incorporated, and further, Johnson teaches restoring into a different type of object development environment the private object state data from the computer-readable file ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6; pg 2 Part 3, Customization section) as claimed.

Per claim 17:

The rejection of claim 16 is incorporated, and further, Johnson teaches an XML structure such that the computer-readable file is configured for being imported both into the object development environment and the different type of object development

environment ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6; pg 2 Part 3, Customization section) as claimed.

Per claim 18:

The rejection of claim 16 is incorporated, and further, Johnson teaches:

creating Java objects based upon the restored private object state data, wherein the Java objects are used within the object development environment ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6) as claimed.

Per claim 19:

The rejection of claim 16 is incorporated, and further, Johnson teaches:

-creating Java objects based upon the restored private object state data ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6; pg 2 Part 3, Customization section)

- storing the Java objects in an object hash table ("Identify the JavaBean's property names, types and values...Instrospector and ...PropertyDescriptor," page 3, part 2)

-retrieving a frame based upon the Java objects stored in the object hash table ("Identify the JavaBean's property names, types and values... Instrospector and ...PropertyDescriptor," page 3, part 2) as claimed.

Per claim 20:

The rejection of claim 1 is incorporated, and further, Johnson teaches:

-restoring the private object state data by processing the computer-readable file ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6) as claimed.

- using the restored private object state data to generate source code ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6; pg 2 Part 3, Customization section) as claimed.

Per claim 21:

The rejection of claim 20 is incorporated, and further, Johnson teaches:

using the restored private object state data to generate a different type of source code ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6; pg 2 Part 3, Customization section) as claimed.

Per claim 22:

The rejection of claim 1 is incorporated, and further, Johnson teaches:

-(a) determining whether the private object state data of the objects have been modified from the initial values given to the objects upon the objects' creation ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6; pg 2 Part 3, Customization section)

-(b) storing in the computer-readable file the private object state data that has been determined in step (a) to have been modified ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6; pg 2 Part 3, Customization section)

-(c) restoring from the computer-readable file the private object state data that has been determined in step (a) to have been modified ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6; pg 2 Part 3, Customization section) as claimed.

Per claim 23:

The rejection of claim 1 is incorporated, and further, Johnson teaches:

-determining customization hooks associated with the objects used within the object development environment ("creates a JavaBean from a DOM tree...within this ...may be

several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6; pg 2 Part 3, Customization section)

-storing the customization hooks in the computer-readable file ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6; pg 2 Part 3, Customization section)

-restoring the customization hooks by processing the computer-readable file ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6; pg 2 Part 3, Customization section).

Per claim 24:

The rejection of claim 1 is incorporated, and further, Johnson teaches:

-determining design time object state data associated with the objects used within the object development environment ("Identify the JavaBean's property names, types and values...Instrospector and ...PropertyDescriptor," page 3, part 2)

-restoring the design time object state data by processing the computer-readable file, wherein the restored design time object state data is used during design time ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6; pg 2 Part 3, Customization section).

Per claim 25:

The rejection of claim 24 is incorporated, and further, Johnson teaches:

-determining run time object state data associated with the objects used within the object development environment ("Identify the JavaBean's property names, types and values...Instrospector and ...PropertyDescriptor," page 3, part 2)

-storing the run time object state data in the computer-readable file ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6; pg 2 Part 3, Customization section).

-restoring the run time object state data by processing the computer-readable file, wherein the restored run time object state data is used during run time ("creates a JavaBean from a DOM tree...within this ...may be several...nodes, each of which corresponds to a property of the JavaBean," page 9, "Creating JavaBeans from XML"; see Figure 3 in Part 2, page 6; pg 2 Part 3, Customization section).

Per claim 32, it is the computer-implemented apparatus version of claim 11, respectively, and is rejected for the same reasons set forth in connection with the rejection of claim 1 above.

Per claim 33:

The rejection of claim 32 is incorporated, and further, Johnson teaches:

Customizing the serialization of an object without requiring a change to the object (pg 2 Part 3, Customization section).

Per claim 34:

The rejection of claim 33 is incorporated, and further, Johnson teaches:

-providing custom hooks for writing out state information of an object (pg 2 Part 3, Customization section) as claimed.

Per claim 35:

The rejection of claim 33 is incorporated, and further, Johnson teaches:

BeanStateInfo object ("Identify the JavaBean's property names, types and values...Instrospector and ...PropertyDescriptor," page 3, part 2) as claimed.

Per claim 36:

The rejection of claim 32 is incorporated, and further, Johnson teaches:

-serializing objects such that only properties that have not changed from their respective default values are written to the computer-readable file ("Identify the JavaBean's property names, types and values...Instrospector and ...PropertyDescriptor," page 3, part 2; "use XML as a serialization format for beans," pg 7 paragraph 2; pg 2 paragraph 6-7 in Part 3; pg 9 last paragraph –first paragraph in pg 10 in Part 3) as claimed.

Response to Arguments

6. Applicant's arguments filed 5/26/2005 have been fully considered but they are not persuasive.

Per claims 11 and 32:

The Applicant states that the Johnson does not disclose or suggest determining that first private object state data is to be restored before second private object state data because of interdependency between them.

In response, Johnson states that the XML document is represented as a tree of objects, whether in Java"...the document has a "hierarchical structure: elements contain other elements..."the <Name> element contains a <First> element and a <Last> element. This "contains" relationship could be represented in a class diagram , as shown in Figure 3...contains references to two other objects...Each of these objects contains other objects...The XML document ...reflects how information relates to other information (page 5, Part 1)." Therefore, the tree structure of XML reflects the "contains" relationship in the document "corresponding to a parent-child relationship in the tree(page 5, Part 1)." The child object is dependent on the parent object, accordingly, the XML tree reflects the interdependency between objects and the document can be processed "as a tree of nodes, instead of as a series of lines or tokens (page 6, Part 1)." Therefore, Johnson discloses the limitation in the claims. If applicant means anything more, this must be brought out in the claims to further clarify the invention.

Applicant's arguments with respect to storing private object state data have been considered but are moot in view of the new ground(s) of rejection.

7. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Insun Kang whose telephone number is 571-272-3724. The examiner can normally be reached on M-F 7:30-4 PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on 571-272-3719. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100.

I. kang
AU 2193

KAKALI CHAKI
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100